UNIVERSITY OF SURREY©

Faculty of Engineering & Physical Sciences

School of Computer Science and Electronic Engineering

Department of Electrical and Electronic Engineering

Postgraduate Programmes in Electrical and Electronic Engineering

EEEM068: Applied Machine Learning

FHEQ Level 7 Examination

Time allowed: 4 hours

Semester 2 2022/3

Answer **ALL** the questions.

Where appropriate the mark carried by an individual part of a question is indicated in square brackets [ ].

Calculators are permitted providing they are non-programmable and not wireless enabled

Additional materials: Open book

Section A

A1.

What are the ways of training a neural network if you do not have sufficient data?

[2 marks]

Solution: Data augmentation techniques such as cropping, padding, and horizontal flipping are commonly used to train large neural networks.

A2.

Mention and explain two reasons for using dropout in a neural network being a good regulariser.

[4 marks]

Solution: Full credit will be awarded if any two of the following reasons are mentioned:
1. Dropout helps prevent feature co-adaptation, which has a regularizing effect.
2. Dropout adds noise to the learning process, and training with noise in general has a regularizing effect.
3. Dropout leads to more sparsity in the hidden units, which has a regularizing effect. (Note that in one of the lectures, this was phrased as dropout "shrinking the weights" or "spreading out the weights". We will also accept this phrasing.)

A3.

(a) What is vanishing gradient?

[4 marks]

Solution: Vanishing gradient is a situation in which a deep multilayer feed-forward network does not have the ability to propagate useful gradient information from the output end of the model back to the layers near the input end of the model.

(b) How is it resolved by residual layer?

[6 marks]

Solution: The residual or skip connections allow gradient information to pass through the layers, by creating "highways" of information, where the output of a previous layer/activation is added to the output of a deeper layer. This allows information from the earlier parts of the network to be passed to the deeper parts of the network, helping maintain signal propagation even in deeper networks.

A4.

Consider the convolutional neural network defined by the layers in the left column below. Fill in the shape of the output volume and the number of parameters at each layer. You can write the activation shapes in the format (H, W, C), where H, W, C are the height, width and channel dimensions, respectively. Unless specified, assume padding 1, stride 1 where appropriate.

Notation:

- CONV-K, N denotes a convolutional layer with N filters with height and width equal to K.
- POOL-K denotes a $K \times K$ max-pooling layer with stride of K and 0 padding.
- FLATTEN flattens its inputs, identical to torch.nn.flatten()
- FC-N denotes a fully connected layer with N neurons.

Note: each row in the following table contains 4 marks.

Layer	Dimension of Feature Map	Number of Parameters
Input	$224 \times 224 \times 3$	0
CONV-3, 16		
ReLU		
POOL-2		
BATCHNORM		
CONV-3, 8		
ReLU		
POOL-2		
FLATTEN		
FC-10		

[36 marks]

Solution:

Layer	Dimension of Feature Map	Number of Parameters
Input	224 x 224 x 3	0
CONV-3, 8	224 x 224 x 16	448
ReLU	224 x 224 x 16	0
POOL-2	112 x 112 x 16	0
BATCHNORM	112 x 112 x 16	32
CONV-3, 16	110 x 110 x 8	3,208
ReLU	110 x 110 x 8	0
POOL-2	55 x 55 x 8	0
FLATTEN	2420	0
FC-10	10	242,010

A5.

(a) Complete the sentence: In a single-layer feed-forward neural network, there is(are) [...] input, [...] hidden and [...] output layer(s) and no [...] connections at all.

[3 marks]

Solution: one, no, one, feedback.

(b) Complete the sentence: Due to the time-consuming nature of computing gradients for each entry in the training corpus, modern DL libraries utilize a technique that gauges the gradient by first randomly sampling a subset from the training corpus, and then averaging only this subset in every epoch. This approach is known as [...]. The actual number of randomly chosen samples in each epoch is termed [...]. The gradient itself is obtained by an algorithm known as [...].

[6 marks]

Solution: stochastic gradient descent. batch size. back propagation.

A6.

In your training loop, you are using SGD and a logistic activation function which is known to suffer from the phenomenon of saturated units.
(a) Explain the phenomenon.

[3 marks]

Solution: The derivative of the logistic activation function is extremely small for either negative or positive large inputs.

(b) You switch to using the tanh activation instead of the logistic activation, in your opinion does the phenomenon still exists?

[4 marks]

Solution: The use of the tanh function does not alleviate the problem since we can scale and translate the sigmoid function to represent the tanh function: $\tanh(z) = 2\sigma(2z) - 1$.

(c) In your opinion, is using the tanh function makes the SGD operation to converge better?

[4 marks]

Solution: While the sigmoid function is centred around 0.5, the tanh activation is centred around zero. Like batch normalization, centring the activations may aid the optimizer converge faster. Note: there is no relation to SGD; the issue exists when using other optimization functions as well.

A7.

Answer the following questions:
(a) Which norm does the following expression represent?

$$|x_1 - x_2| + |y_1 - y_2|$$

[2 marks]

Solution: The L2 norm.

(b) Compute both the Euclidean and Manhattan distance of the vectors:

$$x_1 = [6, 1, 4, 5] \;\&\; x_2 = [2, 8, 3, -1]$$

[8 marks]

Solution:
The Manhattan distance is: $|6-2| + |1-8| + |4-3| + |5-(-1)| = 4 + 7 + 1 + 6 = 18$.

The Euclidean distance is: $\sqrt{(6-2)^2 + (1-8)^2 + (4-3)^2 + \left(5-(-1)\right)^2} = \sqrt{102}$

A8.

State whether the following statement is true or false: *An activation function applied after a Dropout, is equivalent to an activation function applied before a dropout.*

[2 marks]

Solution: True

A9.

You design a binary classifier for detecting the presence of malfunctioning temperature sensors. Non-malfunctioning (N) devices are the majority class in the training corpus. P represents the malfunctioning devices. While running inference on an unseen test-set, you discover that the Confusion Metrics (CM) has the following values:

	Predicted		
Actual		P	N
	P	12	7
	N	24	1009

(a) Find: True Positives (TP), True Negatives (TN), False Positives (FP), False Negatives (FN) and correctly label the numbers in table.

[4 marks]

Solution:

	Predicted		
Actual		P	N
	P	TP = 12	FN = 7
	N	FP = 24	TN = 1009

(b) What is the accuracy of the model?

[4 marks]

Solution: Accuracy = $\frac{12+1009}{12+7+24+1009}$ = 0.97

(c) What is the precision of the model?

[4 marks]

Solution: Precision = $\frac{12}{12+24} = 0.333$

(d) What is the recall of the model?

[4 marks]

Solution: Recall = $\frac{12}{12+7} = 0.631$

Section B
B1.

(a) What is a graph?

[2 marks]

Solution: A graph is a tuple G = (V, E), where V is a set, whose elements are called vertices, and E is a set of paired vertices, whose elements are called edges.

(b) What is an adjacency matrix of a graph?

[4 marks]

Solution: An adjacency matrix of a graph is a square matrix used to represent a finite graph. The elements of the matrix indicate whether pairs of nodes or vertices are adjacent (indicated by 1) or not (indicated by 0) in a graph.

B2.

(a) What is Few-shot learning?

[2 marks]

Solution: Few-shot learning is a type of machine learning where the goal is to learn to recognize new classes with very limited labelled data, typically a small "support set" of examples per class, and sometimes also an additional "query set" of examples for evaluation.

(a) What is query and support set in few-shot learning?

[4 marks]

Solution: The support set is a set of labelled examples (usually just a few) that are used to learn a new class or task. These examples are used by the model during training to learn a representation of the class or task, which can then be used to classify new examples.

The query set is a set of examples used to evaluate the performance of the model after training. The model is evaluated by predicting the class label of each example in the query set and comparing the predictions to the true labels.

B3.

What is the difference between Siamese network and Triplet network?

[6 marks]

Solution: Siamese network and triplet network are both types of deep neural networks used for learning similarity or distance between data points. The main difference between them lies in the number of input examples they use to compute the similarity or distance.

A Siamese network takes in two input examples, and outputs a similarity score or distance metric between them. The network typically consists of two identical sub-networks, each of which takes in one input example and computes a representation of it. The representations are then compared using a similarity or distance function, such as Euclidean distance or cosine similarity, to produce a final output.

On the other hand, a triplet network takes in three input examples: an "anchor" example, a "positive" example that is like the anchor, and a "negative" example that is dissimilar to the anchor. The network computes the distance between the anchor and positive examples, and the distance between the anchor and negative examples, and uses these

B4.

Describe three different methods by which one can fine-tune an ImageNet pre-trained CNN.

[6 marks]

Solution: An ImageNet pre-trained CNN model can be finetuned for the following methods:
(a) Replacing and re-training **only the classifier** (usually the FC layer) of the ImageNet pre-trained CNN, on a target dataset.
(b) Finetuning **all the layers** of the ImageNet pre-trained CNN, on a target dataset.
(c) Finetuning **part of the layers** of the ImageNet pre-trained CNN, on a target dataset.
N.B. These are just a few methods to achieve the goal.

B5.

(a) Compute the derivative of the natural sigmoid function $\sigma(x) = \frac{1}{1+e^{-x}}$

[4 marks]

Solution:

$$\frac{d}{dx}\sigma(x) = \frac{d}{dx}((1+e^{-x})^{-1}) = \left((1+e^{-x})^{(-2)}\right)\frac{d}{dx}(1+e^{-x}) = \frac{e^{-x}}{(1+e^{-x})^2}$$

(b) Establish the relationship between SoftMax and Sigmoid activation functions in the case of mutually exclusive classes.

[6 marks]

Solution: In a classification problem with mutually exclusive classes, where all the values are positive and sum to one, a SoftMax activation function may be used. The SoftMax activation function consists of n terms, such that $\forall \mathrm{i} \in [1, \mathrm{n}]$:

$$f(\mathrm{x}_i) = \frac{e^{\mathrm{x}_i}}{\sum_{j=1}^{K} e^{\mathrm{x}_j}} = \frac{1}{1 + e^{-x_i} \sum_{j \neq i}^{K} e^{x_j}}$$

To compute its partial derivative, we treat all θ_k where, $k \neq i$ as constants and then differentiate θ_i using regular differentiation rules. For a given θ_i, let us define:

$$\beta = \sum_{j \neq i} e^{\mathrm{x}_j}$$

This derives to sigmoid activation function:

$$f(\mathrm{x}_i) = \frac{1}{1 + \beta e^{-x_i}}$$

B6.

Consider a graph neural network (GNN) that is trained on a dataset with 1000 nodes and 2000 edges. The GNN has 3 layers, with hidden layer sizes of 64, 32, and 16, respectively. The input features of each node are 10-dimensional, and the output is binary (0 or 1). The loss function used during training is binary cross-entropy.

(a) What is the total number of parameters in the GNN?

[12 marks]

Solution: The total number of parameters in the GNN can be calculated as follows:
First layer: (10 + 1) * 64 = 704 (10 input features plus 1 bias term, times 64 hidden units).
Second layer: (64 + 1) * 32 = 2080 (64 hidden units from previous layer plus 1 bias term, times 32 hidden units).
Third layer: (32 + 1) * 16 = 528 (32 hidden units from previous layer plus 1 bias term, times 16 hidden units).
Output layer: (16 + 1) * 1 = 17 (16 hidden units from previous layer plus 1 bias term, times 1 output unit). Therefore, the total number of parameters is the sum of the parameters in each layer, which is 704 + 2080 + 528 + 17 = 3329.

(b) What is the time complexity of one forward pass of the GNN?

[4 marks]

Solution: The time complexity of one forward pass of the GNN is $O(|V| * |E| * d)$, where |V| is the number of nodes, |E| is the number of edges, and d is the dimensionality of the input features. In this case, |V| = 1000, |E| = 2000, and d = 10, so the time complexity is O(1000 * 2000 * 10) = O(20,000,000).

(c) If the GNN is trained for 100 epochs with a batch size of 64, what is the total number of training iterations?

[4 marks]

Solution: The total number of training iterations is the number of batches in one epoch times the number of epochs. The number of batches in one epoch is ⌈1000/64⌉ = 16, so the total number of training iterations is 16 * 100 = 1600.

(d) If the validation accuracy of the GNN is 0.8, what is the binary cross-entropy loss on the validation set?

Solution: The validation accuracy of the model is 0.8 means that the GNN correctly classifies 80% of the validation examples. The binary cross-entropy loss is a measure of how well the GNN classifies the validation examples. The lower the loss, the better the GNN performs. The formula for binary cross-entropy loss is:

$$loss = -(y * log(p) + (1 - y) * log(1 - p))$$

where:
y is the ground truth label (0 or 1)
p is the predicted probability of the positive class.

For the validation set, we know that y = 0 for 20% of the examples and y = 1 for 80% of the examples. We also know that p = 0.8 for the 80% of examples where y = 1. The binary cross-entropy loss for the validation set is then:

$$loss = -(0.2 * log(0.2) + 0.8 * log(0.8)) = 0.2$$

[6 marks]

B7.

Low-shot learning is a challenging problem in machine learning, especially when the amount of labelled data is limited. Consider a dataset with 1000 classes and only 10 labelled examples per class. You want to train a low shot learning model to predict the correct class label for unseen examples. You decide to use a prototypical network as your model.

(a) Explain how a prototypical network works and how it can be used for low-shot learning.

[6 marks]

Solution: A prototypical network is a type of metric-learning model that learns a metric space where samples from the same class are close together and samples from different classes are far apart. To train a prototypical network for low-shot learning, we first select a few labelled examples (called "support" examples) from each class as prototypes, and then we learn a metric space where new examples can be classified based on their distance to the prototypes. Specifically, for each episode in the training, we randomly select a few classes and a few support examples per class, and we use them to compute the prototypes for each class. Then we use the prototypes to compute the distances between a query example and each class, and we classify the query example based on the class with the closest prototype.

(b) Suppose you split the dataset into 80% training and 20% validation sets, and you use the validation set to tune the hyperparameters of the prototypical network. What are some hyperparameters you might want to tune, and how would you tune them?

[4 marks]

Solution: Some hyperparameters one might want to tune in a prototypical network include the learning rate, the number of training episodes, the number of support examples per class, and the number of dimensions in the embedding space. To tune the hyperparameters, you could use the validation set to perform a grid search or a random search over a range of values for each hyperparameter and choose the values that give the best validation accuracy.

(c) Suppose you want to evaluate the performance of your prototypical network on a test set. What metric(s) would you use to measure the performance, and how would you interpret the results?

[4 marks]

Solution: To measure the performance of a prototypical network on a test set, one could use the average classification accuracy over all classes. This is computed as the number of correctly classified examples divided by the total number of examples in the test set. A high classification accuracy indicates that the prototypical network can generalize well to new examples from unseen classes.

(d) Suppose you want to improve the performance of your prototypical network by incorporating additional information into the model. What are some ways you might do this, and how might they help?

[4 marks]

Solution: One way to incorporate additional information into a prototypical network is to use auxiliary tasks such as image captioning or object detection to provide additional context about the examples. Another way is to use unsupervised pretraining or transfer learning to initialize the model with features learned from a larger dataset. These approaches may help improve the performance of the prototypical network by providing more information to the model about the examples.

(e) Suppose you want to compare the performance of your prototypical network to that of another low shot learning model. What other model might you choose, and how would you design the comparison experiment?

[4 marks]

Solution: Another low shot learning model one might choose to compare to the prototypical network is a relation network, which learns to compare pairs of examples and output a similarity score. To design the comparison experiment, one could use the same dataset and evaluate both models on the same test set using the same metric. One could also vary the number of labelled examples per class and compare the performance of the two models under different low-shot scenarios.

(f) Suppose you want to investigate the effect of the amount of labelled data on the performance of your prototypical network. What experiments could you perform, and what might you expect to find?

[6 marks]

Solution: To investigate the effect of the amount of labelled data on the performance of the prototypical network, you could perform experiments where you vary the number of labelled examples per class and evaluate the prototypical network on a fixed test set. For example, you could train the prototypical network with 1, 5, or 10 labelled examples per class and measure the classification accuracy on the test set. You might expect to find that the performance of the prototypical network improves as the number of labelled examples per class increases, but that the improvement eventually levels off as the model starts to overfit the training data. You might also find that the performance of the prototypical network is better for some classes than others, depending on the similarity between the examples within each class.

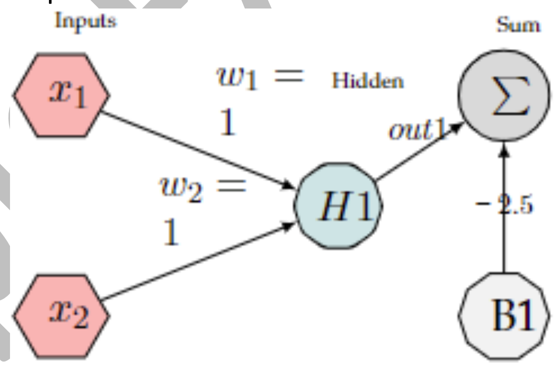
B8.

(a) Explain why in SGD, the number of epochs required to surpass a certain loss threshold increases as the batch size decreases?

[4 marks]

Solution: A larger batch size decreases the variance of the gradient estimation of SGD. Therefore, if your training loop uses larger batches, the model will converge faster. On the other hand, smaller batch sizes increase the variance, leading to the opposite phenomena, longer convergence times.

(b) The following questions refer to the single layer perceptron (SLP) depicted in the following figure. The weights in the SLP are $w_1 = 1$ and $w_2 = 1$ respectively. There is a single hidden node, H1. The bias term, B1 equals −2.5.



Assuming the inputs to the SLP are:

- x1= 0.0 and x2= 0.0
- x1= 0.0 and x2= 1.0
- x1= 1.0 and x2= 0.0
- x1= 1.0 and x2= 1.0

What is the value resulting from the application the sum operator?

[8 marks]

Solution:
The equation of the given SLP is:
$$y = w_1 x_1 + w_2 x_2 + b$$
Where $b = -2.5$
Putting the following values, we obtain the following outputs:

Input	Output (y)
(0,0)	-2.5
(0,1)	-1.5
(1,0)	-1.5
(1,1)	-0.5